Time Series Forecasting Using N-BEATS

Ye Xinyan

CONTENTS

Datasets

-Multivariate multivariate forecasting and univariate time series forecasting using two datasets

Data source: National Bureau of Statistics https://data.stats.gov.cn/
Data Information: China GDP
Time Span: 1980 to 2020
Dataset files: gdp.csv and seasonal_gdp.csv

gdp.csv

Multivariate dataset with original dataset size of (41,28) at year intervals

	Year	第一产业GDP贡献率	第二产业GDP贡献率	第三产业GDP贡献率	就业人员	第一产业就业人员	第二产业就业人员	第三产业就业人员	总人口	0-14岁人口	...	流通中现金(M0)供应量同比增长率(%)	财政收入增长率	居民消费价格指数	商品零售价格指数	工业生产者出厂价格指数	工业生产者购进价格指数	固定资产投资价格指数	全社会固定资产投资	GDP	PGDP
0	1980	-4.8	85.6	19.2	42361.0	29122.0	7707.0	5532.0	98705.0	32384.0	...	29.3	1.2	107.5	106.0	100.5	116.0	108.0	910.9	8.431112	468.0
1	1981	40.5	17.7	41.8	43725.0	29777.0	8003.0	5945.0	100072.0	32384.0	...	14.5	1.4	102.5	102.4	100.2	116.0	108.0	961.0	8.504270	497.0
2	1982	38.6	28.8	32.6	45295.0	30859.0	8346.0	6090.0	101654.0	34146.0	...	10.8	3.1	102.0	101.9	99.8	116.0	108.0	1230.4	8.589216	533.0
3	1983	23.9	43.5	32.7	46436.0	31151.0	8679.0	6606.0	103008.0	32384.0	...	20.7	12.8	102.0	101.5	99.9	116.0	108.0	1430.1	8.702992	588.0
4	1984	25.6	42.7	31.7	48197.0	30868.0	9590.0	7739.0	104357.0	32384.0	...	49.5	20.2	102.7	102.8	101.4	116.0	108.0	1832.9	8.892680	702.0

Datasets

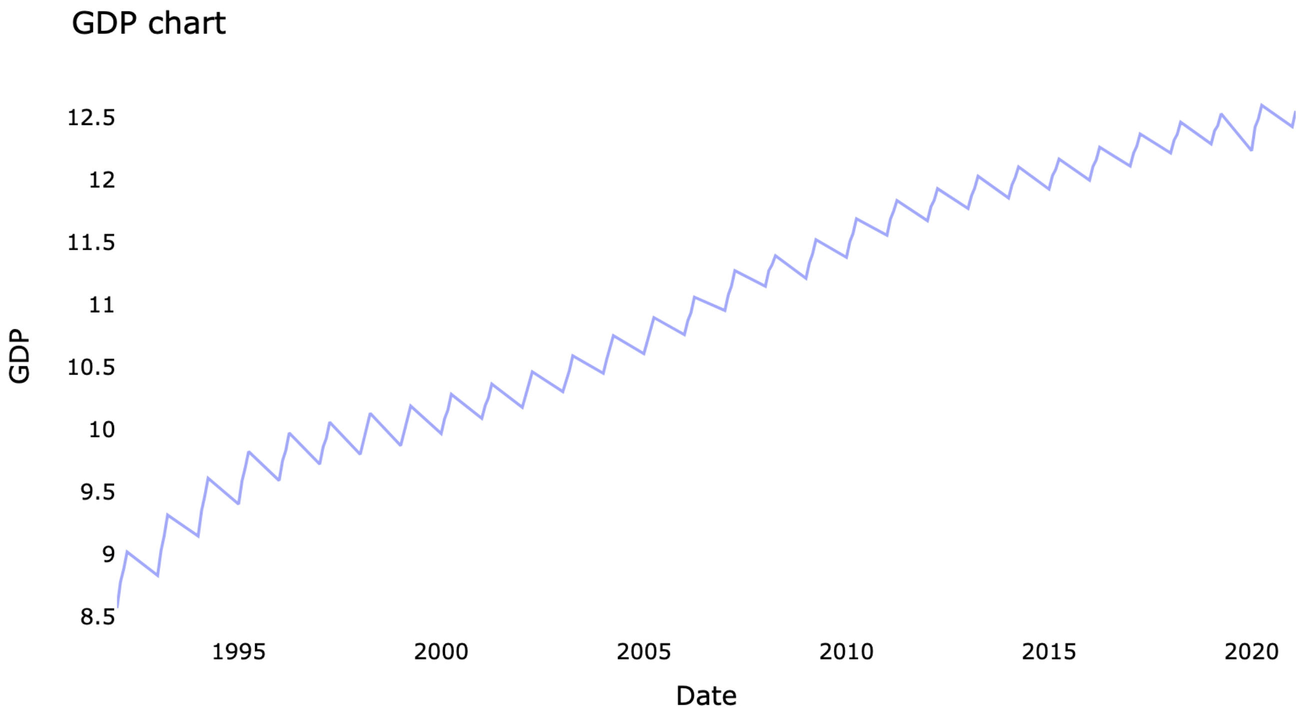
-Multivariate multivariate forecasting and univariate time series forecasting using two datasets

Data source: National Bureau of Statistics https://data.stats.gov.cn/
Data Information: China GDP
Time Span: 1980 to 2020
Dataset files: gdp.csv and seasonal_gdp.csv

seasonal_gdp.csv

Univariate dataset with dataset size (118,2), at quarterly intervals

	Date	GDP
117	1992年第一季度	5262.8
115	1992年第三季度	7192.6
116	1992年第二季度	6484.3
114	1992年第四季度	8254.8
113	1993年第一季度	6834.6
111	1993年第三季度	9385.8
112	1993年第二季度	8357.0
110	1993年第四季度	11095.9



Data Processing

Transform Format

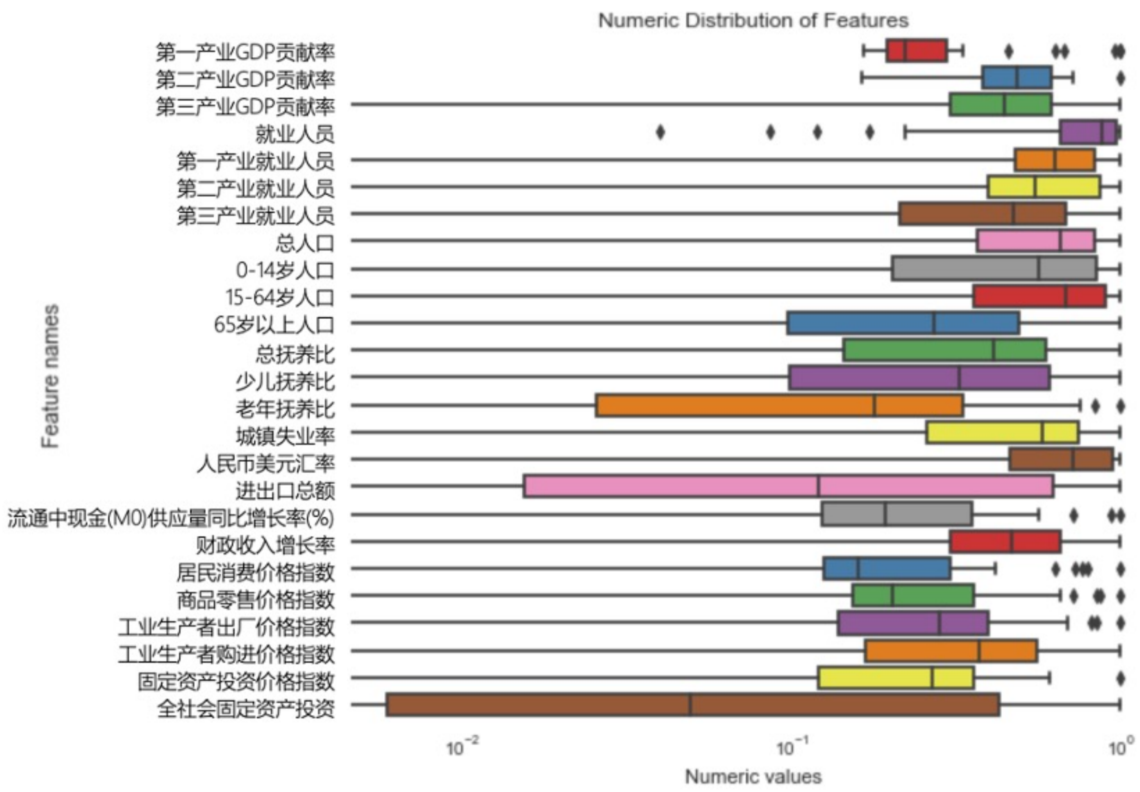
```
from functools import reduce
def str2float(s):
    def dict(s):
        dict1 ={'0': 0, '1': 1, '2': 2, '3': 3, '4': 4,
        return dict1[s]
    def fn(x,y):
        return x*10+y
    a =0
    for i in s :
        a +=1
        if i =='.':
            s = s[:a-1]+s[a:]
            break
    if a ==len(s):
        return reduce(fn,map(dict,s))
    else:
        return reduce(fn,map(dict,s))/(10**(len(s)-a+1))
```

```
def dateformat(date):
    if date[6]=='一':
        return date[:4]+'-01'
    elif date[6]=='二':
        return date[:4]+'-02'
    elif date[6]=='三':
        return date[:4]+'-03'
    elif date[6]=='四':
        return date[:4]+'-04'
    else:
        pass
```

Reconstruction of univariate data with a time window of 10 years, with each preceding 10 years predicting data for the following 10 years

```
def create_dataset(dataset, time_step=1):
    dataX, dataY = [], []
    for i in range(len(dataset)-time_step-1):
        a = dataset[i:(i+time_step), 0]   ###i=0, 0,1,2,3-----99   100
        dataX.append(a)
        dataY.append(dataset[i + time_step, 0])
    return np.array(dataX), np.array(dataY)

# reshape into X=t,t+1,t+2,t+3 and Y=t+4
time_step = 10
X_train, y_train = create_dataset(train_data, time_step)
X_test, y_test = create_dataset(test_data, time_step)
```

Feature Selection



Distribution of data in each feature dimension after scaling using MinMaxScaler

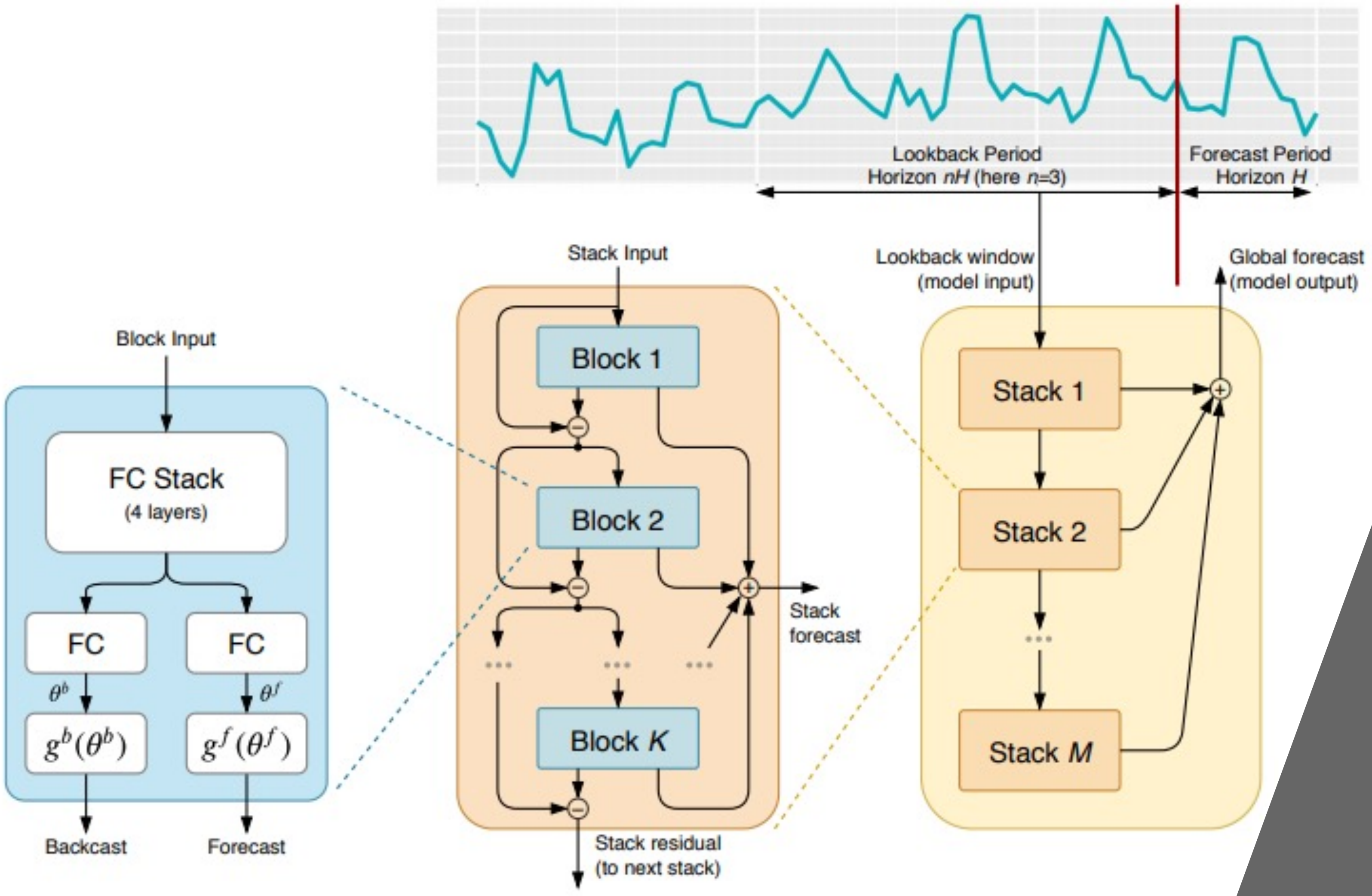
Data Processing

Feature selection

- Feature engineering: MinMaxScaler normalization in advance, then input for feature selection
- Evaluation method: SVR+RandomizedSearchCV on test set, evaluation metric using explained_variance_score

Feature Selection Method	API and Parameters	no. of selected feature (n_feats=26)	Regression Variance (all feats yields 0.91)
Filter_VarianceSelection	VarianceThreshold(threshold=3)	24	0.89
Wrapper_UnivariateSelect	GenericUnivariateSelect(mode='percentile',param=80)	20	0.84
Wrapper_RecursiveFeatsEliminate(DT)	RFECV(DecisionTreeRegressor())	14	0.97
Embedded_MLScore(DT)	SelectFromModel(DecisionTreeRegressor(),threshold='median')	13	0.98
Embedded_MLScore(RF)	SelectFromModel(RandomForestRegressor(),threshold='median')	13	0.96
Embedded_MLScore(GBDT)	SelectFromModel(GradientBoostingRegressor(),threshold='median')	13	0.98
Embedded_MLScore(AdaBoost)	SelectFromModel(AdaBoostRegressor(), threshold='median')	13	0.96

It can be observed that decision trees and GBDT for feature selection can lead to high and stable regression variance, and they are chosen as the feature selection for formal training. It is also found that tree models generally have better feature evaluation capabilities, but like random forests take random sampling and perform inconsistently in each run

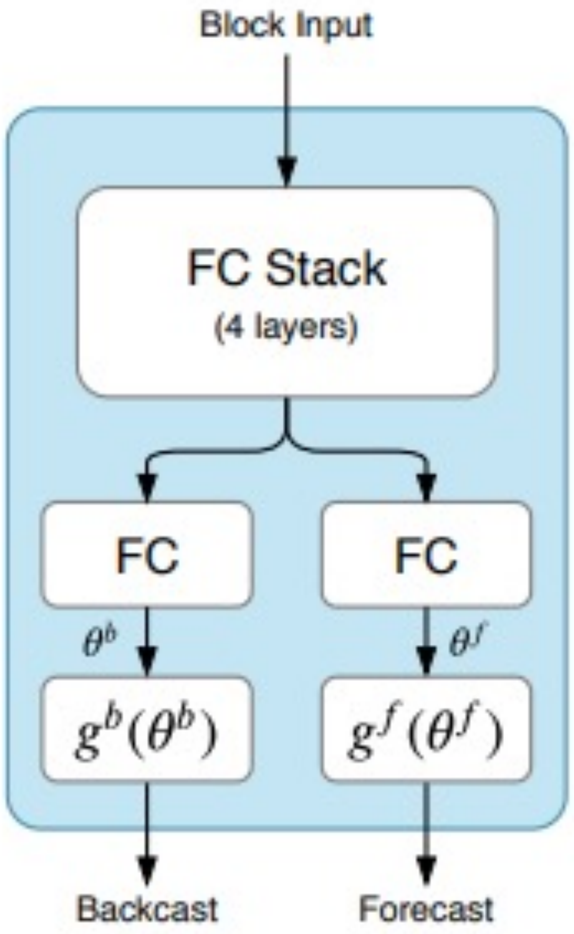


Model Architecture

N-BEATS Model Architecture

To investigate the performance of N-BEATS in time series forecasting problem
The path of the model inputs consists of a backcast period (back horizon) and a forecast period (horizon)

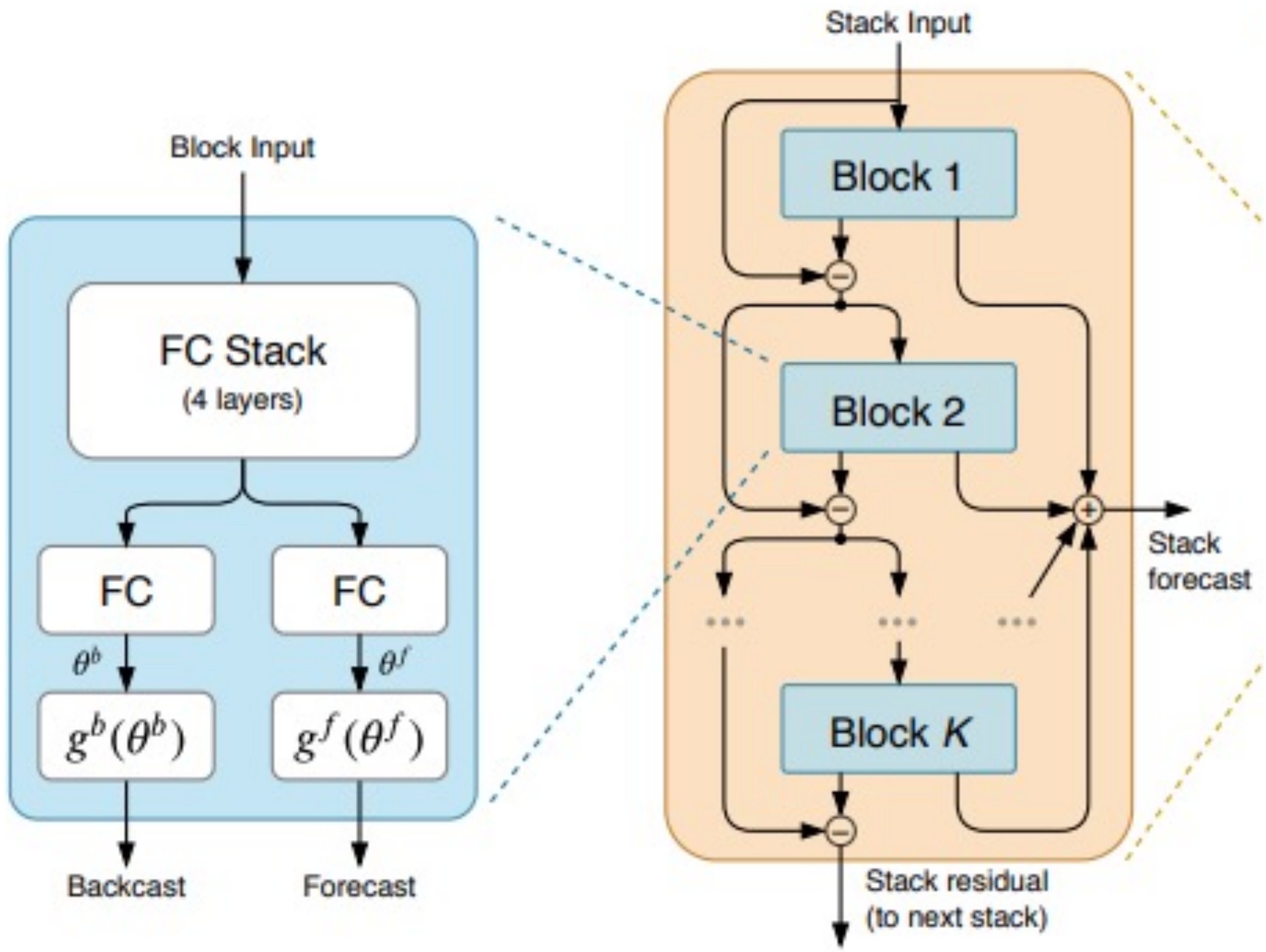
The smallest unit of N-BEATS is the block, with 4 fully connected layers in 1 block, followed by running two more tasks in parallel, one for training θf in the backcast period and the other for predicting θb in the forecast period.
g^f and g^b can be designed to correspond to trend and periodicity. Essentially it is a linear combination of solving for coefficients and basis vectors.



N-BEATS Model Architecture

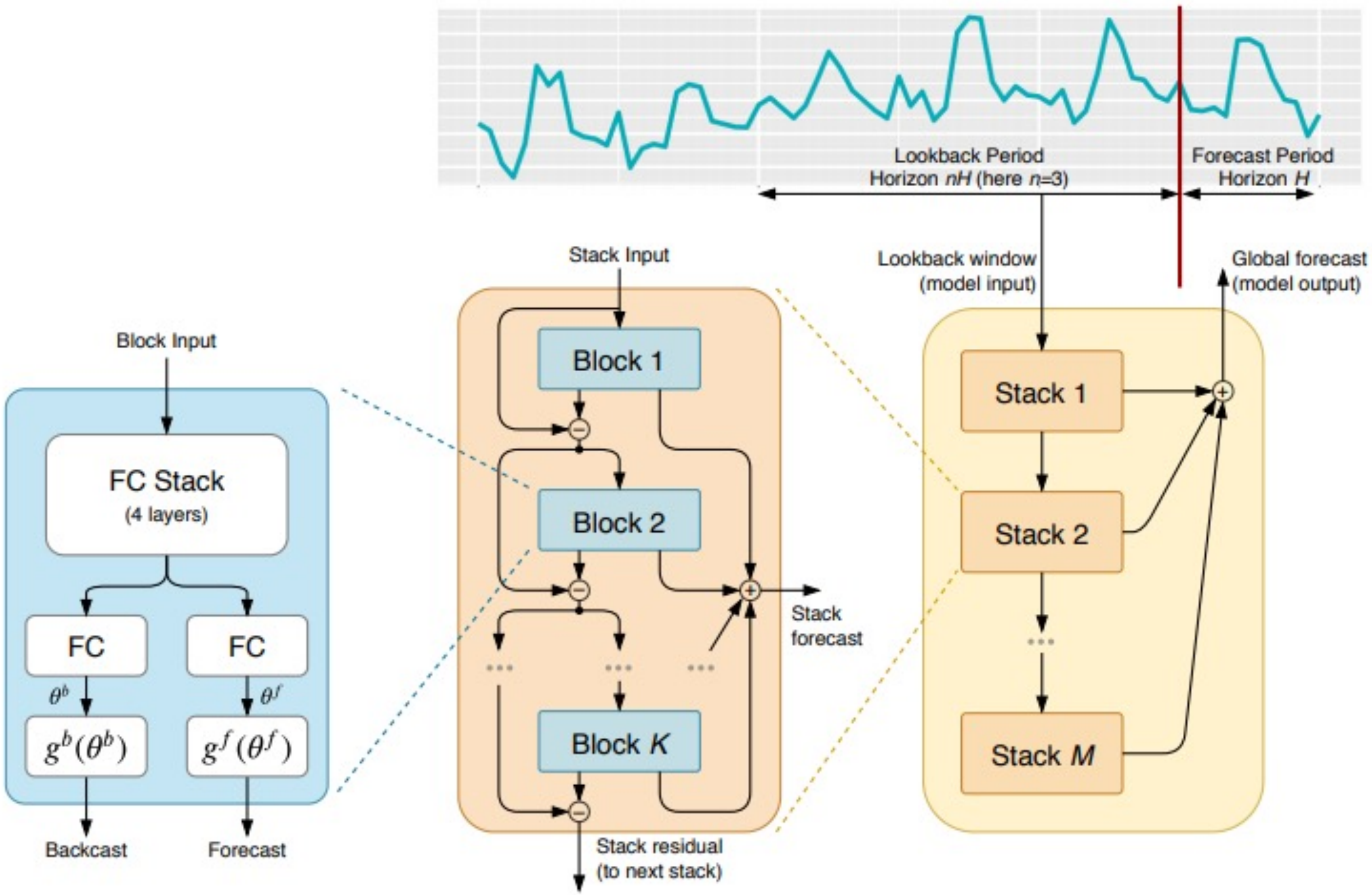
Stack
Multiple blocks are stacked to form a stack, and the blocks are connected by residuals, meaning that the next block aims to learn the residuals from the previous block's transformation, thus preventing subsequent blocks from learning what they learned earlier, and allowing them to focus on the parts that remain unexplained.

N-BEATS uses a dual-residual stacking design, where both backcast and forecast tasks are connected by residuals , and the residuals of the predicted part are aggregated first within the Stack and then across the network.



Boris N. O., Dmitri C., Nicolas C., Yoshua B. N-BEATS: Neural basis expansion analysis for interpretable time series forecasting. arXiv:1905.10437 [cs.LG]

N-BEATS Model Architecture



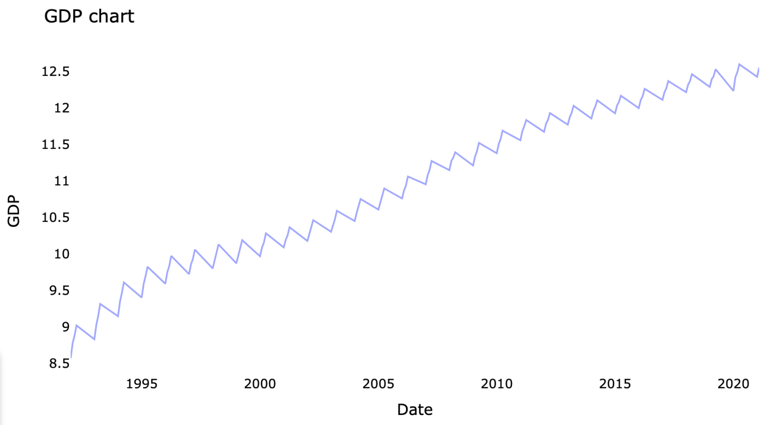
Stacks are also stacked with each other by residuals, aggregated across the network

N-BEATS Performance

Seasonal GDP, univariate time series problem

```
num_samples, time_steps, input_dim, output_dim = 500, 10, 2, 1

# Definition of the model.
model_keras = NBeatsKeras(backcast_length=time_steps, forecast_length=output_dim,
                          stack_types=(NBeatsKeras.GENERIC_BLOCK, NBeatsKeras.GENERIC_BLOCK),
                          nb_blocks_per_stack=2, thetas_dim=(4, 4), share_weights_in_stack=True,
                          hidden_layer_units=64)
# Definition of the objective function and the optimizer.
model_keras.compile(loss='mae', optimizer='adam')

# Train the model.
print('Keras training...')
m=model_keras.fit(X_train, y_train, validation_data=(X_test, y_test), epochs=256, batch_size=128)
```

```
num_samples, time_steps, input_dim, output_dim = 500, 10, 2, 1

# Definition of the model.
model_keras = NBeatsKeras(backcast_length=time_steps, forecast_length=output_dim,
                          stack_types=(NBeatsKeras.TREND_BLOCK, NBeatsKeras.SEASONALITY_BLOCK),
                          nb_blocks_per_stack=2, thetas_dim=(4, 4), share_weights_in_stack=True,
                          hidden_layer_units=64)
# Definition of the objective function and the optimizer.
model_keras.compile(loss='mae', optimizer='adam')

# Train the model.
print('Keras training...')
m=model_keras.fit(X_train, y_train, validation_data=(X_test, y_test), epochs=256, batch_size=128)
```



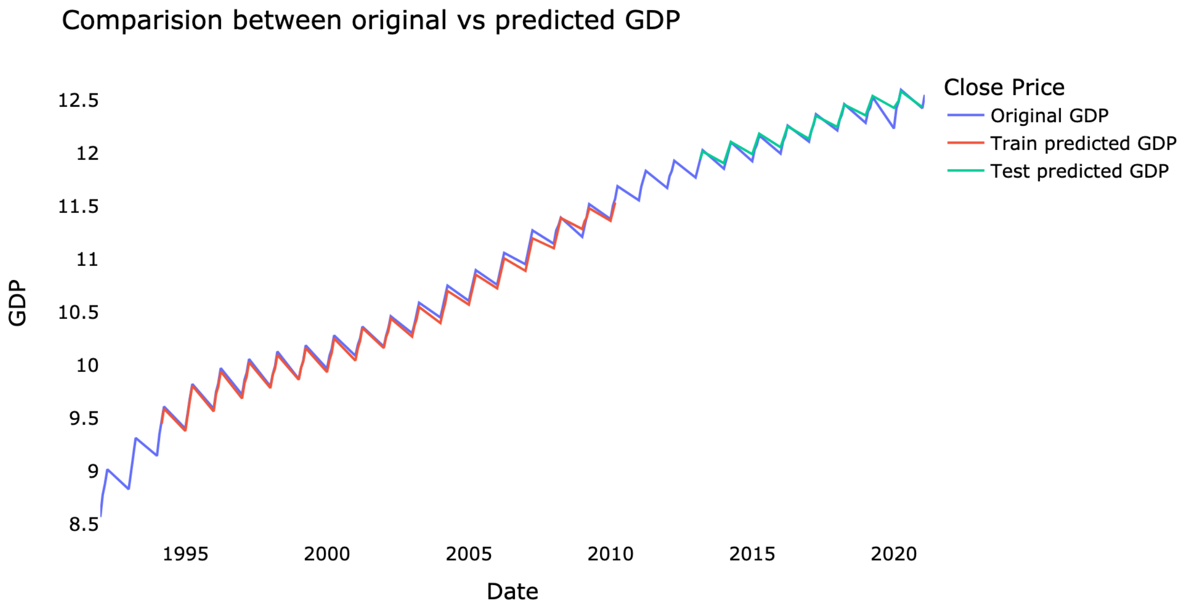
Comparison of stack internal design

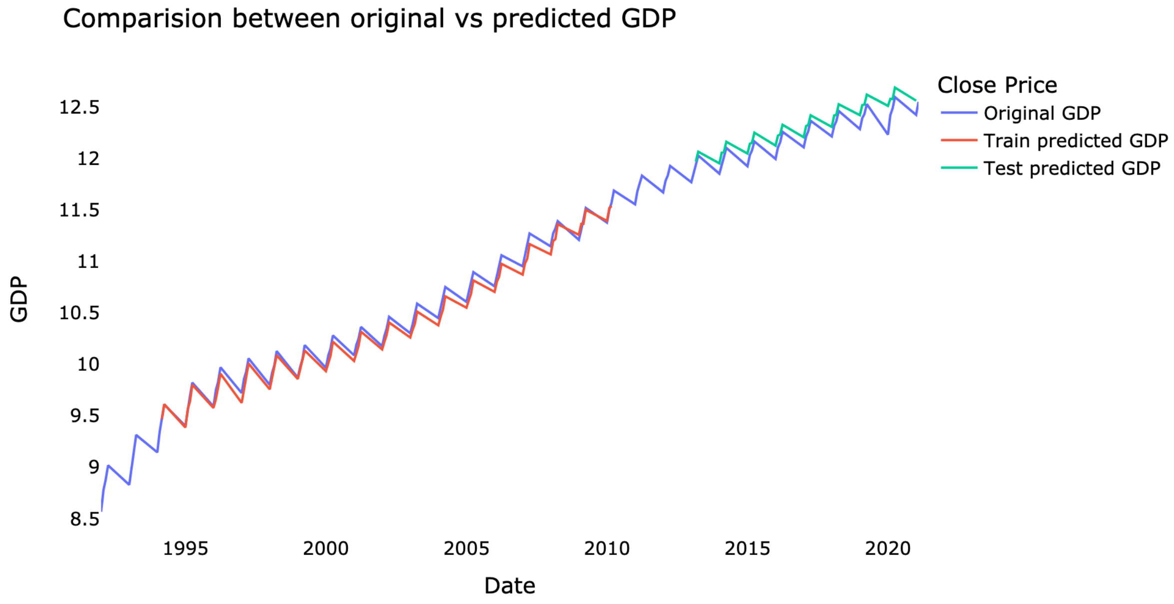
Stack of 2 generic blocks (generic framework, not interpretable)

Stack of 1 trend block and 1 seasonality block (Explainable)

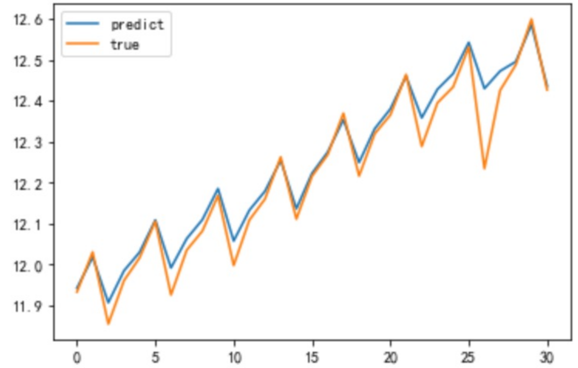
N-BEATS Performance

Seasonal GDP, univariate time





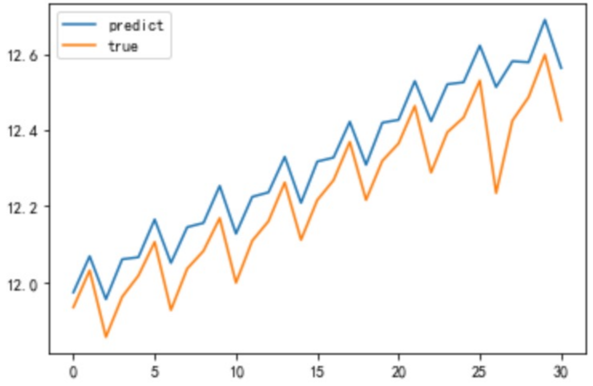
It can be observed that the introduction of an interpretable statistical model for N_BEATS will require a certain degree of accuracy to do trade-offs



Comparison of internal design of the stack

generic framework, non-interpretable

Test data explained variance regression score: 0.963



trend&seasonality interpretable framework

Test data explained variance regression score: 0.948

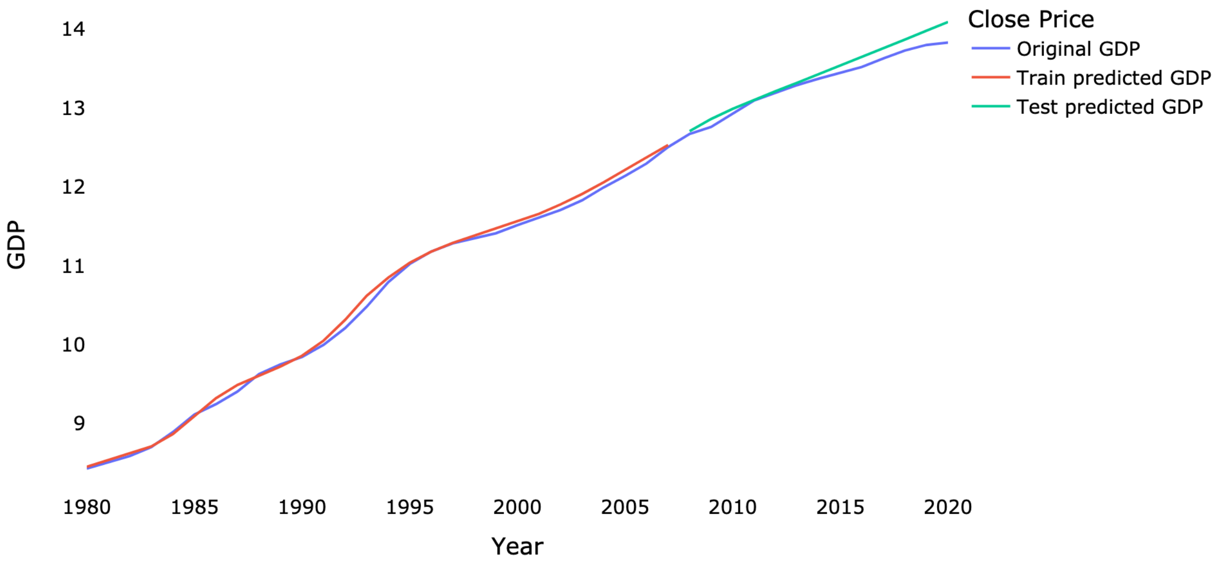
N-BEATS Performance

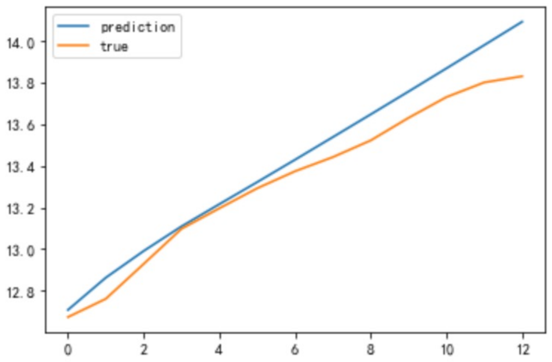
Annual GDP, multivariate forecasting problem, each stack is designed with 2 blocks, trend, seasonality

```
num_samples, time_steps, input_dim, output_dim = 500, 1, 2, 1

# Definition of the model.
model_keras = NBeatsKeras(backcast_length=time_steps, forecast_length=output_dim,
                          stack_types=(NBeatsKeras.TREND_BLOCK,
                           NBeatsKeras.SEASONALITY_BLOCK),
                          nb_blocks_per_stack=2, thetas_dim=(4,4), share_weights_in_stack=True,
                          hidden_layer_units=64)
# Definition of the objective function and the optimizer.
model_keras.compile(loss='mae', optimizer='adam')

# Train the model.
print('Keras training...')
m=model_keras.fit(train_X, train_y, validation_data=(test_X, test_y), epochs=256, batch_size=32)
```

	Year	第一产业GDP贡献率	第二产业GDP贡献率	第三产业GDP贡献率	就业人员	第一产业就业人员	第二产业就业人员	第三产业就业人员	总人口	0-14岁人口	...	流通中现金(M0)供应量同比增长率(%)	财政收入增长率	居民消费价格指数	商品零售价格指数	工业生产者出厂价格指数	工业生产者购进价格指数	固定资产投资价格指数	全社会固定资产投资	GDP	PGDP
0	1980	-4.8	85.6	19.2	42361.0	29122.0	7707.0	5532.0	98705.0	32384.0	...	29.3	1.2	107.5	106.0	100.5	116.0	108.0	910.9	8.431112	468.0
1	1981	40.5	17.7	41.8	43725.0	29777.0	8003.0	5945.0	100072.0	32384.0	...	14.5	1.4	102.5	102.4	100.2	116.0	108.0	961.0	8.504270	497.0
2	1982	38.6	28.8	32.6	45295.0	30859.0	8346.0	6090.0	101654.0	34146.0	...	10.8	3.1	102.0	101.9	99.8	116.0	108.0	1230.4	8.589216	533.0
3	1983	23.9	43.5	32.7	46436.0	31151.0	8679.0	6606.0	103008.0	32384.0	...	20.7	12.8	102.0	101.5	99.9	116.0	108.0	1430.1	8.702992	588.0
4	1984	25.6	42.7	31.7	48197.0	30868.0	9590.0	7739.0	104357.0	32384.0	...	49.5	20.2	102.7	102.8	101.4	116.0	108.0	1832.9	8.892680	702.0

Comparision between original vs predicted GDP



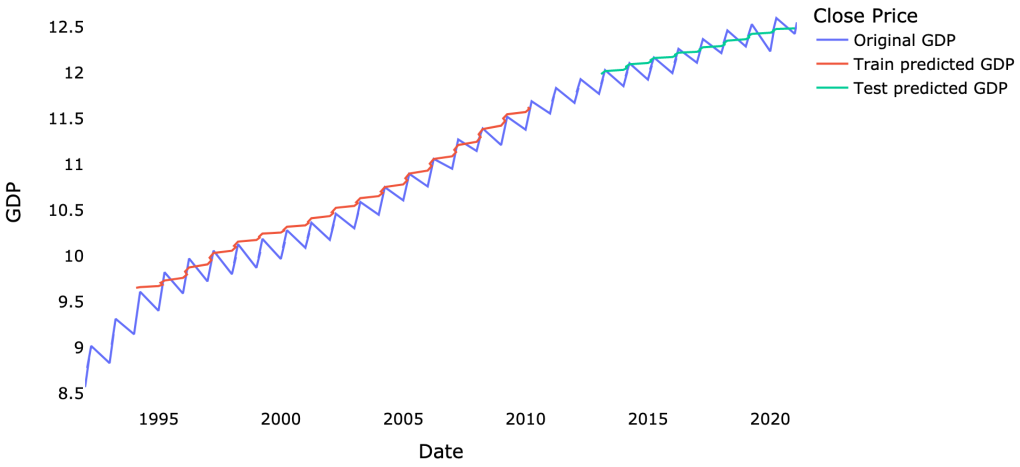


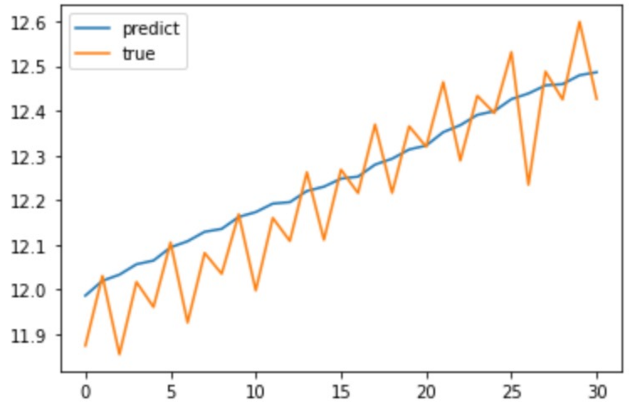
Test data explained variance regression score: 0.971

LSTM Performance

Seasonal GDP, univariate time series problem

Comparision between original vs predicted GDP





Test data explained variance regression score: 0.832

3 * LSTM layer+relu

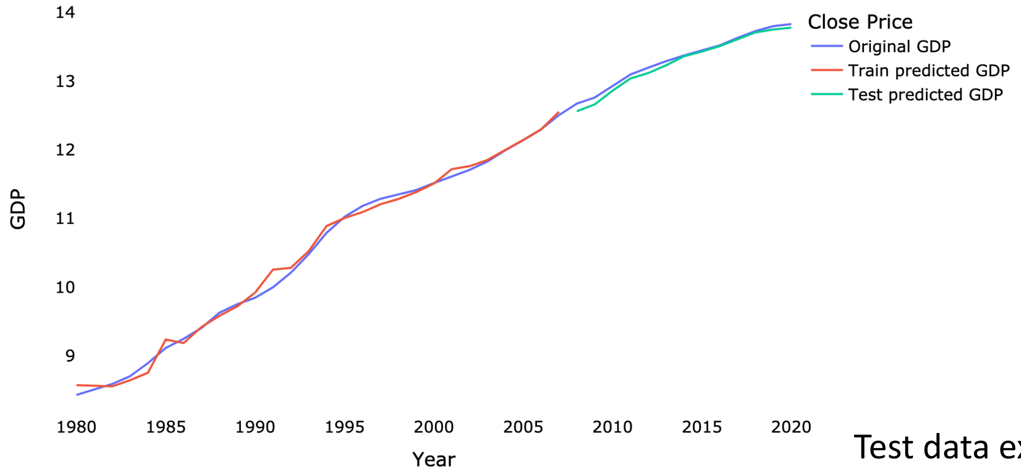
```
def model_builder(hp):
    model=Sequential()
    # Tune the number of units in the LSTM layer
    # Choose an optimal value between 32-512
    hp_units = hp.Int('units', min_value=32, max_value=256, step=32)

    model.add(LSTM(hp_units,return_sequences=True,activation='relu',input_shape=(time_step,1)))
    model.add(Dropout(0.2))

    model.add(LSTM(hp_units,return_sequences=True,activation='relu',dropout=0.2, recurrent_dropout=0.2))
    model.add(Dropout(0.2))

    model.add(LSTM(hp_units,activation='relu',dropout=0.2, recurrent_dropout=0.2))
    model.add(Dropout(0.2))

    model.add(Dense(1,activation='linear'))

    model.compile(loss='mean_squared_error',optimizer='adam')
    return model
```

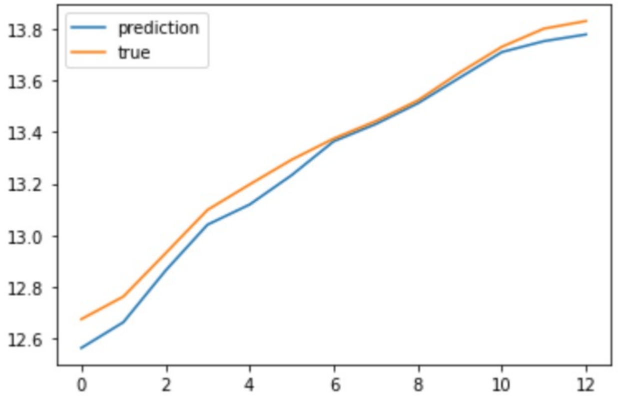
Optimal number of neurons using keras_tuner is 480

```
tuner = kt.Hyperband(model_builder,
                     objective = 'val_loss',
                     max_epochs = 10,
                     factor = 3,
                     directory = 'my_dira',
                     project_name = 'temp')

class ClearTrainingOutput(tf.keras.callbacks.Callback):
    def on_train_end(*args, **kwargs):
        IPython.display.clear_output(wait = True)


tuner.search(X_train, y_train, epochs=10, validation_data=(X_test, y_test),
             callbacks=[ClearTrainingOutput()])
```

Annual GDP, multivariate

Comparision between original vs predicted GDP





Test data explained variance regression score: 0.992

The overall performance of the LSTM in time series prediction is far less than that of N-BEATS, and the tuning process also revealed that increasing the number of LSTM layers beyond 3 resulted in overfitting.

SVM Performance

Annual GDP, multivariate forecasting problem

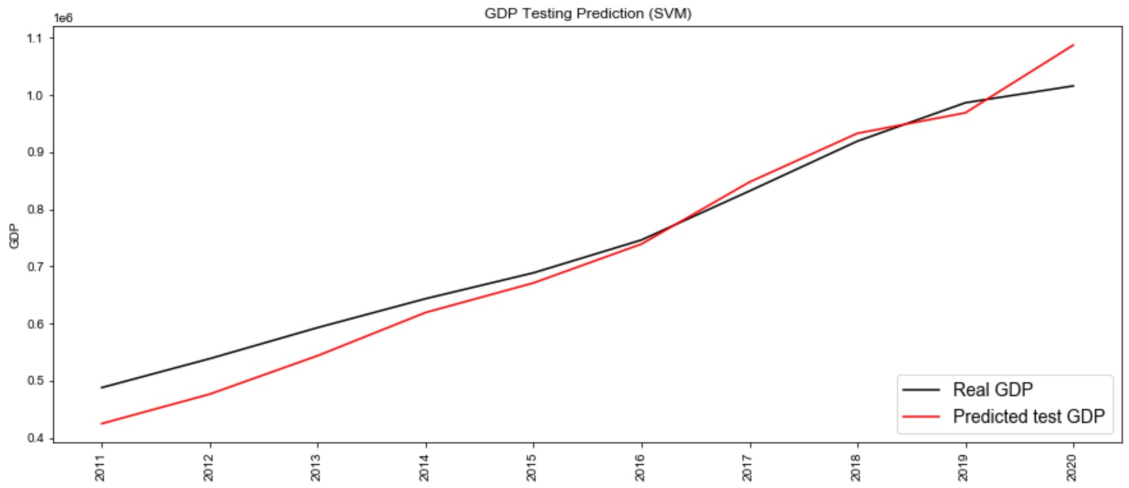
```python
from sklearn.feature_selection import SelectFromModel

DT=DecisionTreeRegressor()
GBDT = GradientBoostingRegressor()
trans = SelectFromModel(GBDT,threshold="0.1*median")
X_trans = trans.fit_transform(X_scaled, y_scaled.ravel())

print("We started with {0} features but retained only {1} of them!".format(X.shape[1], X_trans.shape[1]))
```

```
We started with 26 features but retained only 18 of them!
```

```python
columns_retained_Select = X.columns[trans.get_support()].values
print('----------------------Retained Columns with SelectFromModel are:----------------------')
pd.DataFrame(X_trans, columns=columns_retained_Select).head()
```

```
----------------------Retained Columns with SelectFromModel are:----------------------
```

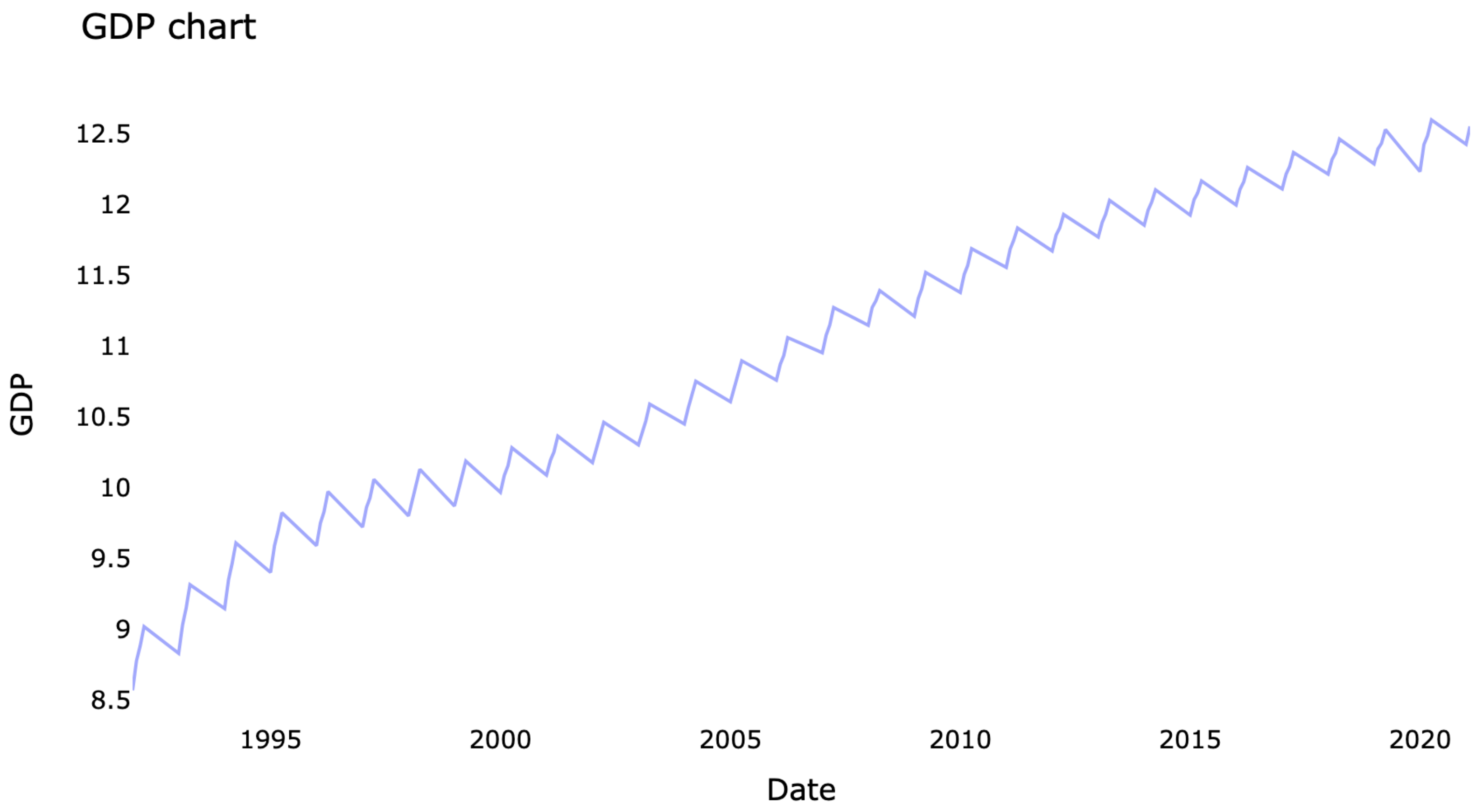
第一产业就业人员	第二产业就业人员	第三产业就业人员	总人口	0-14岁人口	15-64岁人口	65岁以上人口	总抚养比	少儿抚养比	老年抚养比	人民币美元汇率	进出口总额	商品零售价格指数	工业生产者购进价格指数	固定资产投资价格指数	全社会固定资产投资
0.533461	0.000000	0.000000	0.000000	0.851771	0.201225	0.054217	0.721831	0.750769	0.017094	0.000000	0.000000	0.364372	0.555814	0.358621	0.000000
0.564093	0.019073	0.013642	0.032185	0.851771	0.201225	0.054217	0.721831	0.750769	0.017094	0.029016	0.000515	0.218623	0.555814	0.358621	0.000078
0.614694	0.041175	0.018432	0.069432	1.000000	0.000000	0.000000	1.000000	1.000000	0.000000	0.055349	0.000627	0.198381	0.555814	0.358621	0.000496
0.628350	0.062633	0.035476	0.101311	0.851771	0.201225	0.054217	0.721831	0.750769	0.017094	0.067034	0.000904	0.182186	0.555814	0.358621	0.000805
0.615115	0.121335	0.072901	0.133073	0.851771	0.201225	0.054217	0.721831	0.750769	0.017094	0.116372	0.001966	0.234818	0.555814	0.358621	0.001430



Test data explained variance regression score: 0.952

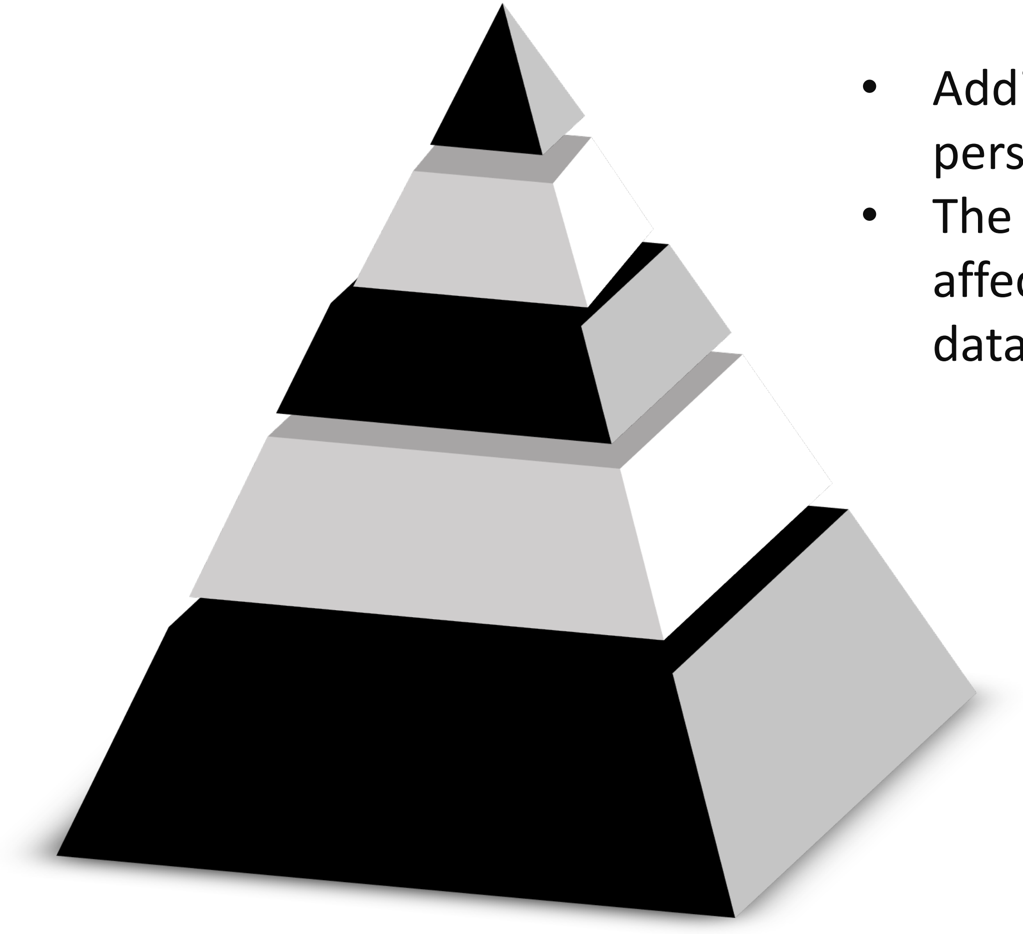
After feature selection Based on GBDT, the input dimension was found that underfitting occurred when 13 features were retained at the median threshold, so the threshold was adjusted to threshold="0.1*median" and 18 features were retained for training

Conclusion



As can be observed from the original data, GDP shows a regular cyclical and year-on-year upward trend. The advantage of using N_BEATS is the ability to incorporate trendiness and periodicity for economic forecasting, not only to effectively learn univariate data with strong periodicity, but also to quickly adapt to multivariate data with smooth trends. In contrast, LSTM is susceptible to overfitting as the number of layers increases, while SVM relies on extensive feature engineering.

Future Work



- Adding the evaluation of regression variance intervals to enhance the persuasiveness of the model performance
- The time dimension could be refined to the month, as GDP itself can be affected by political events or natural disasters, and more granularity of the data can improve the accuracy of the mining